Interview Answers

Interview Questions and Solutions:

1. What is DBSCAN and how does it differ from k-means clustering?

Solution:
DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering algorithm that groups points that are closely packed together and marks points that lie alone in low-density regions as outliers. Unlike k-means, DBSCAN does not require the number of clusters to be specified in advance. It can find arbitrarily shaped clusters and identify outliers, whereas k-means assumes clusters to be spherical and assigns every point to a cluster, even if it's an outlier.

2. Explain the concepts of core points, border points, and noise in DBSCAN.

Solution:

- **Core Points**: These are points that have at least `minPts` within their `ε` (epsilon) neighborhood. They are considered central points of a cluster.
- **Border Points**: These are points that have fewer than `minPts` within their `ε` neighborhood but are in the neighborhood of a core point. They are on the edge of a cluster.
- **Noise Points**: Points that are not core points nor border points. These do not belong to any cluster and are considered outliers or noise.

3. How does DBSCAN handle outliers?

Solution:
DBSCAN treats points that do not belong to any cluster as outliers or noise. These are points that do not meet the criteria to be considered as core or border points. They are not included in any cluster and are typically identified and analyzed separately, making DBSCAN robust to outliers.

4. Discuss the role of `ε` (epsilon) and `minPts` in DBSCAN. How do they affect the clustering?

Solution:

- `ε` (epsilon): This is the radius around each point to look for its neighbors. A larger `ε` makes it easier for points to be considered part of a cluster, leading to fewer, larger clusters. A smaller `ε` can result in more, smaller clusters.
- `minPts`: This is the minimum number of points required to form a cluster. Increasing `minPts` requires more points to be in close proximity to form a dense region, potentially leading to fewer clusters and more noise.

CODE SNIPPET

```python
from sklearn.cluster import DBSCAN
import numpy as np
import matplotlib.pyplot as plt

# Sample dataset: 2D points
X = np.array([[1, 2], [2, 2], [2, 3],
              [8, 7], [8, 8], [7, 7],
              [0, 8], [5, 0], [7, 1], [6, 4]])

# Applying DBSCAN
db = DBSCAN(eps=2, min_samples=2).fit(X)

# Labels for each point
labels = db.labels_

# Plotting the clusters
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')
plt.title('DBSCAN Clustering')
plt.xlabel('X Coordinate')
plt.ylabel('Y Coordinate')
plt.show()
```

Explanation:

- `eps=2`: This sets the neighborhood radius (`ε`) to 2. Points within this distance are considered neighbors.
- `min_samples=2`: This sets the minimum number of points (`minPts`) required to form a dense region to 2. If a point has at least 2 neighbors (including itself)

within its `ε` radius, it's considered a core point.

This example illustrates how changing `eps` and `min_samples` can affect the clustering outcome. Adjusting these parameters can lead to different numbers of clusters and points being marked as noise. Experimenting with these parameters is crucial to achieving meaningful clustering results.